TITLE OF THE INVENTION

A Scalable Network-Attached Storage System

INVENTORS

James O'Reilly

CROSS-REFERENCE TO RELATED APPLICATIONS

None

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

No Federal funds were used to design or develop any parts of this invention.

BACKGROUND OF THE INVENTION

0010 Network-Attached Storage (NAS) systems available in the marketplace today all suffer from a throughput bottleneck caused by their inability to expand the number of computer modules used to move data from storage to the Local Area Network. Industry attempts to resolve this problem hinge around sophisticated file systems, variously described as "global file systems", "distributed file systems" or "locking file systems". All of these methods have so far proved unusable, since the requirement to maintain exactly synchronized copies of the file system impose very severe performance penalties.

0020 This document details a solution using a completely different and novel method of managing the stored data. This method overcomes the bottleneck problem elegantly, permitting not only broad scaling of a NAS system but adding the ability to insert or remove computer modules rapidly and to protect against failure of such modules.

0030 A further elegance of this approach is that it can be applied to many NAS products sold today to add scaling and fault-tolerant features to those products. An enhancement of the invention will allow peer computers (such as Personal Computers and servers) in a network to share their storage in an organized and highly available way, which is a valuable capability since storage drive capacities are becoming very large.

BRIEF SUMMARY OF THE INVENTION

0040 The invention, Scalable Network-Attached Storage (SNAS), addresses a number of problems inherent to current approaches to NAS systems, which are a type of computer system that present storage space to a client user via a Local Area Network (LAN) in such a way that the user sees all or part of the storage as an addressable disk drive

0050 The specific problems addressed by this invention are

1) Scalability of both the storage and the throughput of the LAN interface to that storage, while maintaining high performance,
2) Fail-over of the LAN interface elements,
3) Protection against data loss,
4) Automated response to changes in demand,
5) Creation of Secure-SNAS elements,
6) The creation of Peer-Based Storage Networking, where the rapid growth in the size of disk drives is utilized to allow computers to provide storage to their peer computers.

0060 In contrast to existing approaches, the invention uses a means of dynamic access allocation distribution to achieve solution of the problems, whereby any given computer element is allocated control of a portion of the storage space based on usage or other performance metrics. This means that there is no need to lock or synchronize the file systems in all of the elements, which creates very severe performance problems in the existing approaches.

0070 The invention also provides a means to extend these solutions to create peer-based storage networks. This is a new opportunity, which takes advantage of the extremely rapid growth in storage disk drive capacities that is occurring.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

There are 15 Figures attached to this application. Briefly, these are:

Figure 1: A view of current NAS systems with tightly coupled computer elements, showing the limited scalability of such systems. Here the number of computer elements and data storage elements in the NAS system are severely limited.

Figure 2: A view of current NAS systems using an alternative loosely coupled construction with a shared file system on each computer element. This view shows a typical implementation with a lockbox element configured in the system to attempt to accelerate operations and overcome the severe performance penalty of the shared file system.

Figure 3 shows a preferred implementation the Scalable Network Attached Storage system made possible by this invention. (For drawing purposes the depiction is limited in the number of elements and client computers shown) The figure shows a mixture of shared and exclusively owned data storage elements, all of which are made to appear as a cohesive data set by the SNAS system.

Figure 4 shows a possible installation of the various software elements of the invention on the preferred implementation.

Figure 5 demonstrates the data storage element mapping method for a typical computer system environment, showing a typical example of the three levels of mapping: as the client computer sees the file system; as the SNAS system allocates access to the data storage elements and as implemented on physical story.

Figure 6 is similar to Figure 5 but reflects how a record-oriented database system might be mapped in an SNAS environment. Here the mapping is more flexible, since the primary data element is a relatively small and independent data record.

Figure 7 shows the two-tier mapping system for free space on the SNAS system, where part of the free space is allocated to specific computer elements, and the remainder allocated by the Dynamic Allocation Controller (DAC) function which adds or subtracts free space from the computer elements as controlled by policy rules.

Figure 8 demonstrates the adding of a new (or failure recovered) computer element to the SNAS system. There are two sub-figures, showing a typical element map just as the new computer element is added and then after the DAC has executed a re-mapping to handle the new element, where access control for some data storage elements is moved to the new computer element.

Figure 9 shows how the operation of the backup mapping system might operate to reduce the time that a given data storage element is unavailable, by more rapidly allowing another

computer element to take over the control of those data storage elements formerly handled by the failed computer element.

Figure 10 demonstrates local data replication where the policy rule system initiates a replication operation between one computer element and another. In this case the replication is shown to occur over the Local Area Network, but alternative implementations might use the Storage Area network or other pathways.

Figure 11 similarly shows replication, but here the pathway is through Wide Area Network communications equipment, allowing a copy of data to be replicated on a distant site for added disaster protection.

Figure 12 shows an example of access replication, where more than one computer element is allowed to access any given data element, so permitting a substantial increase in the number of accesses n a given time to that data storage element. In this example, only one computer element is write-enabled to allow the data storage element to be changed, but the SNAS system allows for multiple change-enabled computer elements if the data type or file users allow it.

Figure 13 shows an extension of the SNAS concept whereby a portion of the SNAS might be segregated and communicated with using encryption. Data Encryption Agents (DEA) are shown in a number of the elements including several client computers.

Figure 14 portrays a typical Peer-Based Storage Network, where the SNAS means are extended to allow the unused storage of peer client computers and other computers to be used in concert with the SNAS and Secure-SNAS systems to enhance performance and reduce overall systems costs.

Figure 15 shows how the Peer-Based Storage Network might be employed to protect important data, so allowing a High-Availability User Network to be built. Here multiple copies of an important file are distributed across the network and SNAS system.

Figure 16 shows an alternative construction of a SNAS, Secure-SNAS or Peer-Based Storage Network, where a network switch is used as the vehicle for re-directing accesses from the client computers. Other alternative constructions similar to this might put the re-direction functions in a storage network switch or in a hybrid switch containing both storage and communications networking functions.

DESCRIPTION OF THE INVENTION

Overview

0090 As noted in the Brief Description above, this invention solves a number of problems seen in today's NAS systems. These are:

1) Scalability of both the storage and the throughput of the LAN interface to that storage, while maintaining high performance,
2) Fail-over of the LAN interface elements,
3) Protection against data loss,
4) Automated response to changes in demand,
5) Creation of Secure-NAS elements,
6) The creation of Peer-Based Storage Networking, where the rapid growth in the size of disk drives is utilized to allow computers to provide storage to their peer computers.

0100 NAS systems typically consist of one or more computer elements that interface between the storage disk drives and the LAN. This invention, Scalable Network-Attached Storage (SNAS) resolves the above problems by means of dynamically allocating control of the data storage elements (Drives, logical volumes, folders/directories, files or blocks) to the various computer elements. The allocation process uses a variety of algorithmic policy rules to determine the ownership of each data element, including rules involving the size of the data storage element, frequency of access, type of data stored, level of availability guaranteed to the users, security and disaster backup needs. Control of the allocation process resides in one or more of the computer elements, together with, or instead of, their SNAS functions.

0110 Using the allocation policies, the invention distributes the totality of the storage elements to the various computer elements. This distribution is reviewed on a regular basis and adjusted as required. The invention can be tailored to work with a broad variety of storage file systems and existing NAS software. A key requirement is that a computer element must be able to view that part of the storage allocated for its control, but visibility of any other part of the storage system will not impact on the normal operation of the invention, and in many cases is desirable as a means to speed up recovery processes in the event of a computer element failure.

0120 The invention allows sophisticated segmentation of the storage, whereby sections of the storage can be replicated invisibly to the client user to add protection. This replication can

be local to the storage system, or over a data communications link to protect against disaster loss of information. At the same time, the number of computer elements allocated to a given data element can be adjusted. Adding new computer or storage elements is also simple and automatic, since the allocation policy managers will utilize the new assets upon detection by readjusting the allocations.

0130 This flexibility in managing storage by access is extended to utilize the very large storage elements becoming common on Personal Computers and other computers. A Peer-Based Network Storage solution uses part of the available storage on each computer as a Distributed SNAS. This requires dynamic management of the storage and replication services, which the invention provides. However, the policy rules management element of the invention is extended to manage the physical placement of data on the Distributed SNAS, in like manner to the methods used to place data by policy in the SNAS implementations of the invention.

Comparison with Current Methods

0140 The SNAS invention uses a very different method compared with the methods used in computers or existing NAS systems. In multi-processor computer systems, either a single file system or a global file system is used; depending respectively on whether the structure of the computer cluster is considered tightly coupled or loosely coupled. In the tightly coupled case (see Figure 1), the computers access a single image of the file system kept on the data storage elements, and so the computers behave essentially as a single entity in controlling that storage. The need to maintain a "single computer image" creates a very expensive design that is limited in scalability, difficult to change and less resistant to failures. The alternative loosely coupled architecture (Figure 2) treats the computers as relatively independent units. To maintain the integrity of the shared data storage while allowing any of the computers to access any part of that data storage, the system designer resorts to a shared file system (also described as a distributed or global file system).

0150 The invention is designed using a loosely coupled set of computer elements to take advantage of their much lower cost and higher flexibility (see Figure 3). It differs substantially from current approaches by not requiring a shared file system. Instead the invention achieves data storage integrity and systems scaling by use of software elements that control allocation of

access, which is inherently slow to change. It does not suffer from the performance penalty seen in the loosely coupled system architecture approach to NAS, where a shared file system is used.

0160 The shared file system usually require copies of the file system to be maintained in every computer node, which must be completely synchronized at all times. This causes a great deal of overhead for each operation in accessing data, with throughput being reduced as much as 90%. Some of this type of system uses a central "lockbox", where the status of every file is maintained, to attempt to reduce the performance penalty, but the gains involved are not significant, and the performance loss is even more pronounced as the number of computer elements is increased.

Basic Operation

0170 The invention consists of a number of software elements acting in concert, which are applied to the computer elements to form an SNAS system. In a typical implementation (see Fig. 1), these elements include:

a) Dynamic Allocation Controller (DAC)

0180 This software element controls the allocation of segments of the storage pool to a given computer element. It utilizes a set of policy rules, based on templates as modified by the system administrator; to determine which data storage element (Drives, logical volumes, folders/directories, files or blocks) is controlled by which computer element. The data storage elements do not need to be contiguously mapped, since the architecture of the invention permits fragmentation of an element to the level of individual blocks. The DAC periodically updates the allocation of data storage elements as the metrics that control the policies change.

0190 The DAC also provides the agency for handling the addition or subtraction/failure of either computer elements or data storage elements. In all cases, DAC includes the changes when updating the allocations according to the policies during the update cycles subsequent to the hardware change. A typical system will have a primary DAC, which is active, and a backup DAC on another computer element, which is inactive unless the primary DAC fails.

b) Access Re-Director (ARD)

0200 The ARD acts as the initial connection point for the outside clients. This is the element that a client will contact to make (open) a new file access. The ARD finds the computer element that is actually handling the requested file and sends a redirection method to the client, which then communicates with the computer element directly for data transfers.

0210 ARD has the capability of carrying multiple computer elements for any single data element. This allows scaling of throughput, multiplying the performance of the total SNAS system. Multiple ARD elements are allowed in a system, with one designated as primary, and another as first secondary. The reason for this is explained below.

c) Connection Monitor Agent (CMA)

0220 CMA co-resides with the ARD element. It functions as a tracking agent for accesses to specific data element. As such, CMA provides a number of the metrics for use by the DAC in its policy calculations

d) Throughput Monitor Agent (TMA)

0230 TMA co-resides with the NAS software in each computer element. It tracks performance relative to the capability of the computer element and reports back periodically to the DAC. The TMA also acts as a failure detector, since any unit that fails to update metrics is interrogated and action taken if it fails to respond.

e) Policy Administration Interface (PAI)

0240 PAI is an administrator's interface tool that can sit on any computer connected to the SNAS system. It communicates via an encrypted connection to the DAC element, allowing policies to be modified. It contains a Task Scheduler, which permits events to occur at specific times. This permits sophisticated pricing methods for service quality and performance, since the DAC can schedule increasing the number of elements servicing a client's data, for example.

0250 Other software elements can be added, including a Secure-SNAS capability, automatic Data Replication Agents, and Peer-Based Storage Networking services. These are discussed in the appropriate sections below.

0260 Referring to a typical implementation as depicted in Figure 4, we see the aforementioned software elements deployed on a set of computer elements. To demonstrate the operation of this invention in this typical implementation, consider a client requesting a portion of a file from the NAS system. The following steps take place:

1. The client (PC2 in this example) issues an Open File request to the computer element (CE1) containing the ARD.
2. The ARD determines the computer element (in this case CE3) controlling that file, and transmits that information to PC2.
3. The client PC2 communicates the Open File request to CE3.
4. Ce3 verifies access permission to the requested data and acknowledges PC2's request.
5. PC2 begins to read data from CE3.
6. Upon completion of its accesses, PC2 informs CE3 to Close File.

0270 Note that in Figure 4, some of the data storage elements are shared by several computer elements using a Storage Area Network or similar means of sharing (including industry standards SCSI and i-SCSI, and any of a number of proprietary sharing solutions), while CE5 has exclusive ownership of its storage and CE4 has both shared ownership and exclusive ownership. This might occur when existing NAS installations are expanded with the scalable solution provided by this invention. In a situation like this, the access control for the exclusive storage elements would reside with the computer element to which they are attached, but the benefits of having a single policy control method and a virtual pool of storage when looking from the client end of the system provide strong advantages to the approach. This is a very important factor when the invention is expanded to provide a policy-driven Peer-Based Storage Network, since in this case many of the data storage elements will be exclusively attached to clients.

0280 Figure 5 shows how the typical set of data storage elements is mapped. On the top is the client's hierarchical view of the storage. This view sees a large single drive that is fragmented into folders, and then sub-folders then files, as in a typical file system. These folders are allocated to the various computer elements, as determined by the DAC software, resulting in every data storage element having at least one computer element referencing it. Finally, the

allocated data storage elements are mapped onto physical storage devices, as shown on the bottom of Figure 5.

0290 A database, table, record type system as found in relational databases can also be supported by the architecture of SNAS. This is schematically shown in Figure 6, where it can be seen that the database, table and record elements replace the drive, folder/directory file elements. In this case, the database software and/or its associated operating system and software interface drivers may need a modification to allow data allocation to take place. Block level transfer systems such as i-SCSI can also be served by a version of the invention that responds to the i-SCSI operating protocol. In this case, sets of data blocks become the data storage elements.

0300 Operations that increase or decrease the amount of storage used are more complicated in the type of system described herein than in a "standard" file system. The pool-of-storage concept underlying this invention implies central ownership of all the free space in the data storage elements. Though workable, this centralization presents a potential performance bottleneck, so in the preferred implementation the DAC uses a two-tier approach to free-space maintenance. Each mapped element has an associated free space allocated, so that many decisions to use free-space elements or return used elements to the free-space pool can be taken at the computer element level, rather than at the DAC level. A protocol is established where the computer element will request the DAC to make a change in free space allocated to its control whenever the computer element detects that its free space is outside the range of system policies.

0310 Those operations that create or delete data storage elements can require additional steps to keep the mapping of data storage elements intact. The Create or Delete requests fall into one of two categories. In the simpler form, a request is received by a computer element and the affected entity is totally under the control of the computer element. An example of this is where a computer element completely controls a directory and is requested to delete a sub-directory. Here the Create or Delete operation proceeds without additional steps and is executed directly by the computer engine first receiving the request. If however the control of that parent data storage element were split between several computer elements and the data sub-element (the sub-directory in the example) were in fact controlled by another computer element, it would be necessary for the first computer element, which manages the parent element, to regain control of that sub-element prior to deletion. This is done by a request to the DAC, which re-maps the sub-element to the parent data storage element's computer element. Once the computer element

controls the whole data storage element affected, it can make the deletion. When multiple computer elements manage a data sub-element the process is repeated for each computer element until there is a single owner at the parent element level.

Scaling The Elements

0320 The access allocation method described herein allows for very broad scaling of both data storage and throughput to the LAN. This is a direct result of an architecture where there is no need for real-time synchronization between file tables on many computer elements, which has been shown to cause performance losses of over 90%. As a result, NAS systems, which are currently either limited to two computer elements or that offer low performance in scalability, can be superseded by SNAS systems that are able to scale to very large numbers of computer elements, perhaps into the hundreds. It is worth noting that the independence of scaling between the data storage elements and the computer elements provides a major improvement in system administration.

0330 Adding data storage elements is very simple with this system. Upon a computer element detecting that a new storage element is present, the DAC is notified and the new space added to the free-space pool.

0340 New computer elements are also easy to add (Figure 7), since each new element will signify its presence through the TMA interface to the DAM. The DAC then applies its allocation policies to the expanded pool of computer elements, thereby balancing the load over the new elements.

0350 Removal of an element is more complex. In the case of voluntary removal, the data on a data storage element is first copied to other elements using a utility tool. Voluntary removal of a computer engine requires the DAC to be notified so that the data storage elements allocated to the computer element that is being removed are available on other computer elements. In both cases, the request to remove can be achieved by a variety of methods including the PMI or other management software or by a physical method such as a pushbutton on the element.

Fail-over and Recovery

0360 Involuntary removal or failure of an element creates the same challenges as in most computer systems. The easiest to deal with is the loss of a computer element. The TMA process

detects this loss, when the failed element does not provide its regularly scheduled performance metrics. Recovery is relatively simple. The DAC rebuilds the access allocation map with the failed unit removed and the client operations timeout and retry to the new unit.

0370 To minimize loss of data access and aborted client operations, pre-configuring an inactive backup computer element for each element that can quickly rebuild the file structure substantially speeds up recovery (Figure 9). This inactive backup will normally be on a computer element that is actively controlling other data elements, though a dedicated backup is supported in the architecture. When a distributed backup system is used, the inactive backups are identified for all the active computer elements in such a way as that the failure of both active and inactive elements is very unlikely. The backups are mapped over the set of active computer elements to achieve this, so that typically each computer element has an active and a backup role. Since the inactive backups require minimal computer resources, the invention allows for multiple backups of any computer element. The instantaneous increase in load when the inactive allocation is brought on stream will be resolved relatively rapidly, as the TMA reports high loading to the DAC element, which then re-allocates the loading in the next few mapping cycles, load-balancing the system.

Data Protection

0380 Protection against data corruption or loss in the storage require the same mechanisms as in most systems, including the use of RAID arrays and backup storage. A feature of the access allocation type system is a Data Replication Agent software element on each computer element that makes local and remote copies of any data storage element and/or backups according to policies downloaded from the DAM. This is shown in Figure 10. Remote copies provide protection against a local disaster that might destroy or damage any copies in the local area (Figure 11).

0390 When a data storage element is modified by a computer element, the DRA checks to see if a copy of that new data is needed. If required, the DRA initiates a change to the appropriate replica or replicas of the file being changed. This is executed as a transaction between computer elements. The policy system may associate a variety of priorities to this replication, including immediate replication, delayed replication and the sequence in which the replication occurs within the total system.

Access Replication

0400 Another dimension of scaling follows from the concept of an inactive backup for a computer element. In many applications, a small set of files is very actively accessed. In this case, it is possible that a single computer element might prove incapable of supporting client demand. By making the inactive backup active (see Figure 12), it is possible to effectively double throughput, though, for simplicity it is necessary to restrict write operations to just one node. With this restriction, it is, in fact, possible to have a large number of computer elements capable of addressing any give data storage element.

0410 Access replication of this type can be used to minimize loss of availability of a section of the data storage elements through failure of a computer element, since the replicated computer elements can immediately service reissued requests from those clients originally communicating with the failed computer element.

Secure-SNAS Systems

0420 In any NAS system, unauthorized access to data is a serious concern. At a minimum, system administrators can access the files stored in a NAS system, and often users are very poor with password security. To overcome this, part of the structure of this invention supports a secure environment, which we are calling Secure-SNAS (Figure 13). This environment is like the SNAS environment provided by this invention, but takes advantage of the invention's ability to segregate access to generate a sub-set of the storage that is encrypted at source by the user's client computer or user's server computer, and that can be accessed only by a sub-set of the computer elements of the invention.

0430 As shown in Figure 4, the Secure-SNAS and non-secure SNAS environments can co-exist with a Data Encryption Agent (DEA) being installed in those computers requiring encrypted storage capability. An extension to the DEA permits the user to establish a file policy system on that user's computer to determine whether any given file or data storage element should be replicated locally or remotely, and if and how it is encrypted. Additionally, the DEA provides the capability for encrypting command and control flows, including character padding on short messages, and for encrypting file structure information. The DEA may use hardware encryption assist capabilities if available.

Peer-Based Storage Networks

0440 As the data storage capacities of hard disk drives have grown, the amount of storage unused in a network of Personal Computers or other computers has grown substantially, reaching levels comparable with large NAS storage systems. The invention described herein makes such a Peer-Based Storage Network (PBSN) possible by providing a vehicle for distributing storage across the data storage elements in such a network under the control of policy rule sets (See Figure 14).

0450 Each computer member of the network has a PBSN agent installed. This agent allows the user to set up a space in the computer's storage that can be shared with other users. The PBSN agent then sends the address and size information to the DAC via the LAN, and the DAC maps the computer into the PBSN as a computer element, and the defined storage as data storage elements. From this point, operation continues as in the NAS system developed from this invention and described above.

0460 Should the user wish to disconnect some part or all of the storage, or extend the size of that storage, the PBSN agent provides an interface to the DAC. In that case where the storage is reduced, the DAC will move some or all of the stored files to other computers in the PBSN.

0470 In this type of a network, security is a crucial issue. Normally the information stored in the PBSN space is encrypted at source using a Secure-SNAS element provided with the PBSN agent. This means the originator of that data has privacy control over it. Where the computer operating system supports the capability, the PBSN space on a computer is inaccessible directly by the users of that computer and/or is hidden from their view in normal operation.

0480 A PBSN can extend to multiple remote locations, with policies controlling the placement of data at such remote locations. For example, all the important data at a site might be duplicated at several of the other sites and vice versa. In this case, too, replication policies can provide a local version of a remote site's files automatically, so speeding up access to shared data.